BEST AVAILABLE IMAGES

Defective images within this document are accurate representations of the original documents submitted by the applicant.

Defects in the images may include (but are not limited to):

- BLACK BORDERS
- TEXT CUT OFF AT TOP, BOTTOM OR SIDES
- FADED TEXT
- ILLEGIBLE TEXT
- SKEWED/SLANTED IMAGES
- COLORED PHOTOS
- BLACK OR VERY BLACK AND WHITE DARK PHOTOS
- GRAY SCALE DOCUMENTS

IMAGES ARE BEST AVAILABLE COPY.

As rescanning documents *will not* correct images, please do not report the images to the Image Problem Mailbox.

Amendments to the Claims:

This listing of claims replaces all prior versions and listings of claims in the application:

Listing of Claims:

1. (Original) A computer instruction comprises:

a move and duplicate instruction that causes a processor to load a first portion of bits of a source into a first portion of a destination register and duplicate that first portion of bits in a subsequent portion of the destination register.

2. (Original) The instruction of claim 1 in which the first portion of the source is 64-bits representing a double floating point data type in a memory location.

3. (Original) The instruction of claim 1 in which the first portion of the source is 64-bits representing a double floating point data type in a source register.

4. (Original) The instruction of claim 1 in which the first portion of the destination register is loaded with bits [63-0] of the first portion of the source and the subsequent portion of the destination register is loaded with bits [63-0] of the first portion of the source.

5. (Original) A method comprising:

in a processor, loading a first portion of bits of a source into a first portion of a destination register; and

duplicating the first portion of bits in a subsequent portion of the destination register.

6. (Original) The method of claim 5 in which the first portion of the source is 64-bits representing a double floating point data type in a memory location.

7. (Original) The method of claim 5 in which the first portion of the source is 64-bits representing a double floating point data type in a source register.

8. (Original) The method of claim 5 in which the first portion of the destination register is loaded with bits [63-0] of the first portion of the source and the subsequent portion of the destination register is loaded with bits [63-0] of the first portion of the source.

9. (Original) A computer program product residing on a computer readable medium having instructions stored thereon which, when executed by the processor, cause the processor to:

load a first portion of bits of a source into a first portion of a destination register; and

duplicate the first portion of bits in a subsequent portion of the destination register.

10. (Original) The computer program product of claim 9 in which the first portion of the source is 64-bits representing a double floating point data type in a memory location.

11. (Original) The computer program product of claim 9 in which the first portion of the source is 64-bits representing a double floating point data type in a source register.

12. (Original) The computer program product of claim 9 in which the first portion of the destination register is loaded with bits [63-0] of the first portion of the source and the subsequent portion of the destination register is loaded with bits [63-0] of the first portion of the source.

13. (Original) A computer instruction comprises:

a move one double floating point and duplicate instruction that causes a processor to load 64-bits of a source and return the 64-bits in a lower half of a destination and a upper half of a destination.

14. (Original) The instruction of claim 13 further comprising:

a source operand; and

a destination operand.

15. (Original) The instruction of claim 13 in which the source operand is a memory location.

16. (Original) The instruction of claim 15 in which the memory location has a 128-bit value that represents a double floating point data type.

17. (Original) The instruction of claim 13 in which the source operand is a 128-bit source register.

18. (Original) The instruction of claim 17 in which the source register has a 128-bit value that represents a double floating point data type.

19. (Original) A method executed in a processor comprising:
loading a first number N of bits from a source into a lower half of a 2N wide-bit destination register and in a upper half of the 2N-bit wide destination register.

20. (Original) The method of claim 19 in which the source is a memory location and where N is 64 bits.

21. (Original) The method of claim 20 in which the memory location contains a double floating point data type.

22. (Original) The method of claim 19 in which the source is a 128-bit source register and N is 64 bits.

23. (Original) The method of claim 19 in which the 128-bit source register contains a double floating point data type.

24. (Original) A computer program product residing on a computer readable medium having instructions stored thereon which, when executed by the processor, cause the processor to:

load 64-bits from a source in a lower half of a 128-bit destination register and in an upper half of the 128-bit destination register.

25. (Original) The computer program product of claim 24 in which the source is a memory location containing a 128-bit double floating point data type.

26. (Original) The computer program product of claim 24 in which the source is a 128-bit source register containing a 128-bit double floating point data type.

27. (Original) A computer instruction comprises:

a move packed single floating point high and duplicate instruction that causes a processor to load bits [127-0] of a source and return bits [63-32] of the source in bits [31-0] of a 128-bit destination register, bits [63-32] of the source in bits [63-32] of the destination register, bits [127-96] of the source in bits [95-64] of the destination register and bits [127-96] of the source in bits [127-96] of the destination register.

28. (Original) The instruction of claim 27 further comprising:

a source operand field; and

the destination operand field.

29. (Original) The instruction of claim 27 in which the source operand is a memory location.

30. (Original) The instruction of claim 29 in which the memory location has 128-bits representing a packed single floating point data type.

31. (Original) The instruction of claim 27 in which the source operand is a 128-bit source register.

32. (Original) The instruction of claim 31 in which the source register has 128-bits representing a packed single floating point data type.

33. (Original) A method executed in a processor comprising:

accessing bits [127-0] of a source; and

returning bits [63-32] of the source in bits [31-0] and bits [63-32] of the destination register; and

bits [127-96] of the source in bits [95-64] and bits [127-96] of the destination register.

34. (Original) The method of claim 33 in which the source is a memory location.

35. (Original) The method of claim 34 in which the memory location contains a packed single floating point data type.

36. (Original) The method of claim 33 in which the source is a 128-bit source register.

37. (Original) The method of claim 36 in which the 128-bit source register contains a packed single floating point data type.

38. (Original) A computer program product residing on a computer readable medium having instructions stored thereon which, when executed by the processor, cause the processor to:

load bits [127-0] of a source;

return bits [63-32] of the source in bits [31-0] of a 128-bit destination register;

return bits [63-32] of the source in bits [63-32] of the destination register;

return bits [127-96] of the source in bits [95-64] of the destination register; and

return bits [127-96] of the source in bits [127-96] of the destination register.

39. (Original) The computer program product of claim 38 in which the source is a memory location.

40. (Original) The computer program product of claim 39 in which the memory location contains a packed single floating point data type.

41. (Original) The computer program product of claim 38 in which the source is a 128-bit source register.

42. (Original) The computer program product of claim 41 in which the 128-bit source register contains a packed single floating point data type.

43. (Original) A computer instruction comprises:

a move a packed single floating point low and duplicate instruction that causes a processor to load bits [127-0] of a source and return bits [31-0] of the source in bits [31-0] of a 128-bit destination register, bits [31-0] of the source in bits [63-32] of the destination register, bits [95-64] of the source in bits [95-64] of the destination register and bits [95-64] of the source in bits [127-96] of the destination register.

44. (Original) The instruction of claim 43 further comprising:

a source address field; and

the destination register.

45. (Original) The instruction of claim 44 in which the source is a memory location.

46. (Original) The instruction of claim 45 in which the memory location contains 128-bits representing a packed single floating point data type.

47. (Original) The instruction of claim 43 in which the source is a 128-bit source register.

48. (Original) The instruction of claim 47 in which the source register contains 128-bits representing a packed single floating point data type.

49. (Original) A method comprising:

in a processor, loading bits [127-0] of a source;

returning bits [31-0] of the source in bits [31-0] of a 128-bit destination register;

returning bits [31-0] of the source in bits [63-32] of the destination register;

returning bits [95-64] of the source in bits [95-64] of the destination register; and

returning bits [95-64] of the source in bits [127-96] of the destination register.

50. (Original) The method of claim 49 in which the source is a memory location.

51. (Original) The method of claim 50 in which the memory location contains a packed single floating point data type.

52. (Original) The method of claim 51 in which the source is a 128-bit source register.

53. (Original) The method of claim 52 in which the 128-bit source register contains a packed single floating point data type.

54. (Original) A computer program product residing on a computer readable medium having instructions stored thereon which, when executed by the processor, cause the processor to:

load bits [127-0] of a source;

return bits [31-0] of the source in bits [31-0] of a 128-bit destination register;

return bits [31-0] of the source in bits [63-32] of the destination register;

return bits [95-64] of the source in bits [95-64] of the destination register; and

return bits [95-64] of the source in bits [127-96] of the destination register.

55. (Original) The computer program product of claim 54 in which the source is a memory location.

56. (Original) The computer program product of claim 55 in which the memory location contains a packed single floating point data type.

57. (Original) The computer program product of claim 54 in which the source is a 128-bit source register.

58. (Original) The computer program product of claim 57 in which the 128-bit source register contains a packed single floating point data type.

New claims:

59. (New) A hardware-based multithreaded processor comprising:

a plurality of microengines, each of the microengines comprising:

a control store;

controller logic;

context event switching logic; and

an execution box data path including an arithmetic logic unit (ALU) and a general purpose register set, the ALU performing functions in response to instructions, one of the instructions causing the ALU to load a first portion of bits of a source into a first portion of a destination register and duplicate that first portion of bits in a subsequent portion of the destination register.

60. (New) The processor of claim 59 in which the first portion of the source is 64-bits representing a double floating point data type in a memory location.

61. (New) The processor of claim 59 in which the first portion of the source is 64-bits representing a double floating point data type in a source register.

62. (New) The processor of claim 59 in which the first portion of the destination register is loaded with bits [63-0] of the first portion of the source and the subsequent portion of the destination register is loaded with bits [63-0] of the first portion of the source.

63. (New) A hardware-based multithreaded processor comprising:

a plurality of microengines, each of the microengines comprising:

a control store;

controller logic;

context event switching logic; and

an execution box data path including an arithmetic logic unit (ALU) and a general purpose register set, the ALU performing functions in response to instructions, one of the instructions causing the ALU to load a first number N of bits from a source into a lower half of a 2N wide-bit destination register and in a upper half of the 2N-bit wide destination register.

64. (New) The processor of claim 63 in which the source is a memory location and where N is 64 bits.

65. (New) The processor of claim 64 in which the memory location contains a double floating point data type.

66. (New) The processor of claim 63 in which the source is a 128-bit source register and N is 64 bits.

67. (New) The processor of claim 63 in which the 128-bit source register contains a double floating point data type.

68. (New) A hardware-based multithreaded processor comprising:

a plurality of microengines, each of the microengines comprising:

a control store;

controller logic;

context event switching logic; and

an execution box data path including an arithmetic logic unit (ALU) and a general purpose register set, the ALU performing functions in response to instructions, one of the instructions causing the ALU to access bits [127-0] of a source, return bits [63-32] of the source in bits [31-0] and bits [63-32] of the destination register, and bits [127-96] of the source in bits [95-64] and bits [127-96] of the destination register.

69. (New) The processor of claim 68 in which the source is a memory location.

70. (New) The processor of claim 69 in which the memory location contains a packed single floating point data type.

71. (New) The processor of claim 68 in which the source is a 128-bit source register.

72. (New) The processor of claim 71 in which the 128-bit source register contains a packed single floating point data type.